Mapping URLs to Razor Pages using routing

Keywords:

"MVC design pattern and how ASP.NET Core uses it to generate the UI for an application using Razor Pages"

"page handlers that act as mini controllers for a request"

"calls the application model to retrieve or save data"

"selecting which Razor Page to invoke in response to a given request"

"endpoint routing system introduced in ASP.NET Core 3.0"

"separation routing can bring between the layout of your Razor Page files and the URLs you expose"

"use routing with Razor Pages to create dynamic URLs"

"build powerful route templates"

"use the routing system to generate URLs"

"decouples your Razor Pages from the underlying URLs"

"customize the conventions Razor Pages uses"

"change the built-in conventions, such as using lowercase for your URLs"

"routing as the glue that ties the middleware pipeline to Razor Pages and the MVC framework"

Questions:

Q: What design pattern does ASP.NET Core use to generate the UI for an application using Razor Pages?

A: The MVC design pattern.

Q: What do page handlers in Razor Pages act as?

A: They act as mini controllers for a request.

Q: What is the process called for selecting which Razor Page to invoke in response to a request?

A: Routing.

Q: What routing system was introduced in ASP.NET Core 3.0?

A: The endpoint routing system.

Q: What is one benefit of using routing with Razor Pages?

A: It allows a single Razor Page to handle requests to multiple URLs.

Q: How does using URL generation benefit Razor Pages?

A: It decouples your Razor Pages from the underlying URLs that are used to execute them.

Q: What does customizing conventions in Razor Pages allow you to do?

A: It gives you complete control over the URLs your application uses.

Q: What built-in convention can be changed using customization?

A: Using lowercase for your URLs.

Q: What does routing tie together in an ASP.NET Core application?

A: The middleware pipeline to Razor Pages and the MVC framework.

What is routing?

Keywords:

"Routing is the process of mapping an incoming request to a method that will handle it"

"control the URLs you expose in your application"

"automatically extracting data from a request’s URL"

"middleware pipeline, which defines the behavior of your application"

"cross-cutting concerns, such as logging and error handling"

"Endpoint-Middleware at the end of your middleware pipeline"

"page handler on a Razor Page or an action method on an MVC controller"

"process of mapping a request to a handler is called routing"

"handler is a page handler method in a Razor Page"

"Even a simple URL path—for example, /Index—uses routing"

"file layout-based routing system"

"Use a different Razor Page for every product"

"query string is part of a URL that contains additional data that doesn’t fit in the path"

"query string and routing approaches are very similar"

"completely customize the URLs"

"routing fundamentally decouples the URLs in your application from the filenames of your Razor Pages"

"modify your exposed URLs without having to change your Razor Page filenames or locations"

"default URLs are normally the best option in the long run"

Questions:

Q: What is routing in ASP.NET Core?

A: Routing in ASP.NET Core is the process of mapping an incoming HTTP request to a specific handler.

Q: What does routing allow you to control in your application?

A: The URLs you expose in your application.

Q: What can routing automatically extract from a request’s URL?

A: Data.

Q: What defines the behavior of an ASP.NET Core application?

A: The middleware pipeline.

Q: What kind of concerns is middleware well suited to handle?

A: Cross-cutting concerns, such as logging and error handling.

Q: What middleware is typically used to handle more complex application logic?

A: The Endpoint-Middleware at the end of your middleware pipeline.

Q: What can the Endpoint-Middleware invoke to handle a request?

A: A page handler on a Razor Page or an action method on an MVC controller.

Q: What determines which Razor Page or action method is executed for a request?

A: Routing.

Q: In Razor Pages, what is the handler for a request?

A: A page handler method in a Razor Page.

Q: How is the Razor Page Index.cshtml executed from a simple URL like /Index?

A: Using routing.

Q: What is a limitation of using a purely file layout-based routing system in an e-commerce app?

A: You'd need a different Razor Page for every product in your product range.

Q: What is the purpose of the query string in a URL?

A: It contains additional data that doesn't fit in the path.

Q: How are the query string and routing approaches similar to Razor Pages?

A: The Razor Page dynamically displays the results for the correct product as appropriate.

Q: What advantage does routing provide over query strings for product URLs?

A: You can completely customize the URLs.

Q: What does routing decouple in your application?

A: The URLs from the filenames of your Razor Pages.

Q: How can you change the URL for a Razor Page without changing its filename or location?

A: By modifying your exposed URLs depending on your routing configuration.

Q: What does the author say about the necessity of customizing URLs?

A: The default URLs are normally the best option in the long run.

Routing in ASP.NET Core

Keywords:

"Routing has been a part of ASP.NET Core since its inception"

"in ASP.NET Core 3.0 it went through some big changes"

"routing was restricted to Razor Pages and the ASP.NET Core MVC framework"

"no dedicated routing middleware in your middleware pipeline"

"routing happened only within Razor Pages or MVC components"

"cross-cutting concerns, like authorization"

"inevitable duplication, which wasn’t ideal"

"a new routing system was introduced, endpoint routing"

"makes the routing system a more fundamental feature of ASP.NET Core"

"no longer ties it to the MVC infrastructure"

"Razor Pages and MVC still rely on endpoint routing"

"other middleware can use it too"

"ASP.NET Core 5.0 uses the same endpoint routing system as ASP.NET Core 3.0"

"convention-based routing and attribute routing"

"how routing works for Razor Pages"

Questions:

Q: When did routing go through big changes in ASP.NET Core?

A: In ASP.NET Core 3.0 it went through some big changes.

Q: Where was routing restricted to in ASP.NET Core 2.0 and 2.1?

A: Routing was restricted to Razor Pages and the ASP.NET Core MVC framework.

Q: What was missing from the middleware pipeline in ASP.NET Core 2.0 and 2.1?

A: There was no dedicated routing middleware in your middleware pipeline.

Q: Where did routing happen in earlier versions of ASP.NET Core?

A: Routing happened only within Razor Pages or MVC components.

Q: What made some things messy when routing was restricted to MVC infrastructure?

A: Some cross-cutting concerns, like authorization, were restricted to the MVC infrastructure and were hard to use from other middleware.

Q: What was a downside of restricting routing to the MVC infrastructure?

A: That restriction caused inevitable duplication, which wasn’t ideal.

Q: What new system was introduced in ASP.NET Core 3.0?

A: A new routing system was introduced, endpoint routing.

Q: What does endpoint routing do in ASP.NET Core?

A: A more fundamental feature of ASP.NET Core.

Q: What no longer ties the routing system to the MVC infrastructure?

A: Endpoint routing.

Q: Who can use endpoint routing besides Razor Pages and MVC?

A: Other middleware can use it too.

Q: What routing system does ASP.NET Core 5.0 use?

A: ASP.NET Core 5.0 uses the same endpoint routing system as ASP.NET Core 3.0.

Q: What types of routing are available in ASP.NET Core?

A: Convention-based routing and attribute routing.

Q: What does this section cover about Razor Pages?

A: How routing works for Razor Pages.

Using endpoint routing in ASP.NET Core

Keywords:

"Endpoint routing is fundamental to all but the simplest ASP.NET Core apps"

"EndpointMiddleware—You use this middleware to register the endpoints in routing the system"

"executes one of the endpoints at runtime"

"EndpointRoutingMiddleware—This middleware chooses which of the endpoints registered by the EndpointMiddleware should execute"

"referring to this middleware as the RoutingMiddleware"

"configure all the endpoints in your system"

"register your Razor Pages and MVC controllers"

"health-check endpoints that confirm your application is still running"

"An endpoint in ASP.NET Core is some handler that returns a response"

"Each endpoint is associated with a URL pattern"

"Razor Page handlers and MVC controller action methods"

"use simple middleware as an endpoint"

"call UseEndpoints in the Configure method of Startup.cs"

"automatically register all the Razor Pages in your application using extensions such as MapRazorPages"

"register other endpoints explicitly using methods such as MapGet"

Questions:

Q: What is fundamental to all but the simplest ASP.NET Core apps?

A: Endpoint routing is fundamental to all but the simplest ASP.NET Core apps.

Q: What does EndpointMiddleware do at runtime?

A: The middleware executes one of the endpoints at runtime.

Q: What is the role of EndpointRoutingMiddleware?

A: This middleware chooses which of the endpoints registered by the EndpointMiddleware should execute for a given request.

Q: How is EndpointRoutingMiddleware referred to throughout the book?

A: As the RoutingMiddleware.

Q: What is configured in the EndpointMiddleware?

A: All the endpoints in your system.

Q: What kinds of handlers can be registered in EndpointMiddleware besides MVC controllers?

A: Additional handlers that fall outside of the MVC framework, such as health-check endpoints.

Q: What is an endpoint in ASP.NET Core?

A: An endpoint in ASP.NET Core is some handler that returns a response.

Q: What is each endpoint associated with?

A: Each endpoint is associated with a URL pattern.

Q: What typically makes up the bulk of the endpoints in an application?

A: Razor Page handlers and MVC controller action methods.

Q: What else can be used as an endpoint besides controller actions or Razor Pages?

A: You can also use simple middleware as an endpoint, or a health-check endpoint.

Q: How do you register endpoints in your application?

A: Call UseEndpoints in the Configure method of Startup.cs.

Q: How can all Razor Pages in your application be registered automatically?

A: Using extensions such as MapRazorPages.

Q: What method can be used to register other endpoints explicitly?

A: Using methods such as MapGet.

Registering multiple endpoints in Startup.Configure

Keywords:

"Each endpoint is associated with a route template that defines which URLs the endpoint should match"

"A route template is a URL pattern that is used to match against request URLs"

"The EndpointMiddleware stores the registered routes and endpoints in a dictionary"

"RoutingMiddleware compares an incoming request to the routes registered in the dictionary"

"attaches that to the request's HttpContext object"

"EndpointMiddleware checks to see which endpoint was selected and executes it"

"If the request URL doesn't match a route template, the RoutingMiddleware doesn't select an endpoint"

"the EndpointMiddleware silently ignores the request and passes it to the next middleware in the pipeline"

"typically a 404 response is returned when the request reaches the dummy 404 middleware"

"Only middleware placed after the RoutingMiddleware can detect which endpoint is going to be executed"

"AuthorizationMiddleware is placed after the RoutingMiddleware"

"can access certain metadata about the endpoint"

"must be placed after the RoutingMiddleware and before the EndpointMiddleware in your middleware pipeline"

"make sure you place it after the RoutingMiddleware and before the EndpointMiddleware"

"how the RoutingMiddleware and EndpointMiddleware interact to provide routing capabilities in ASP.NET Core"

Questions:

Q: What defines which URLs an endpoint should match?

A: Each endpoint is associated with a route template that defines which URLs the endpoint should match.

Q: What is a route template?

A: A route template is a URL pattern that is used to match against request URLs.

Q: Where does the EndpointMiddleware store the registered routes and endpoints?

A: In a dictionary.

Q: What does the RoutingMiddleware do at runtime with incoming requests?

A: It compares an incoming request to the routes registered in the dictionary.

Q: What happens if RoutingMiddleware finds a matching endpoint?

A: It makes a note of which endpoint was selected and attaches that to the request's HttpContext object.

Q: What does the EndpointMiddleware do when a selected endpoint is present?

A: The middleware checks to see which endpoint was selected and executes it.

Q: What happens if the request URL doesn't match a route template?

A: The RoutingMiddleware doesn't select an endpoint, but the request still continues down the middleware pipeline.

Q: What does the EndpointMiddleware do if no endpoint is selected?

A: It silently ignores the request and passes it to the next middleware in the pipeline.

Q: What response is typically returned when the request reaches the dummy middleware?

A: Typically a 404 response is returned when the request reaches the dummy 404 middleware.

Q: Which middleware can detect which endpoint will be executed?

A: Only middleware placed after the RoutingMiddleware can detect which endpoint is going to be executed.

Q: Where is the AuthorizationMiddleware placed in the middleware pipeline?

A: The AuthorizationMiddleware is placed after the RoutingMiddleware.

Q: What can the AuthorizationMiddleware access about the endpoint?

A: It can access certain metadata about the endpoint, such as its name and what the required permissions are.

Q: Where must the AuthorizationMiddleware be placed in the middleware pipeline?

A: After the RoutingMiddleware and before the EndpointMiddleware.

Q: What must you ensure if your middleware needs to know which endpoint will be executed?

A: You need to make sure you place it after the RoutingMiddleware and before the EndpointMiddleware.

Q: What does the RoutingMiddleware and EndpointMiddleware interaction provide?

A: Routing capabilities in ASP.NET Core.

Convention-based routing vs. attribute routing

Keywords:

"Routing is a key part of ASP.NET Core"

"Using global, convention-based routing"

"Using attribute routing"

"Convention-based routing is defined globally for your application"

"your MVC controllers must adhere strictly to the conventions you define"

"attribute-based routes to tie a given URL to a specific endpoint"

"\[Route] attributes on the action methods themselves"

"you can explicitly define what the URL for each action method should be"

"Razor Pages uses conventions to generate attribute routes"

"you get the predictability and terseness of convention-based routing with the easy customization of attribute routing"

"attribute routing is the best option and is the recommended approach"

"you’ll define your application’s expected URLs using route templates"

"Route templates define the structure of known URLs in your application"

"A segment is a small contiguous section of a URL"

"you can define Specific, expected strings"

"Constraints on segments of a URL, such as ensuring that it’s numeric"

Questions:

Q: What is a key part of ASP.NET Core?

A: Routing is a key part of ASP.NET Core.

Q: What are the two different ways to define URL-endpoint mappings in ASP.NET Core?

A: Using global, convention-based routing and using attribute routing.

Q: What determines which routing approach you should use?

A: Whether you’re using Razor Pages or MVC controllers, and whether you’re building an API or a website.

Q: How is convention-based routing defined?

A: Convention-based routing is defined globally for your application.

Q: What is a downside of convention-based routing?

A: It makes customizing the URLs for a subset of controllers and actions more difficult.

Q: How are attribute-based routes defined in MVC controllers?

A: By placing \[Route] attributes on the action methods themselves.

Q: What is an advantage of attribute routing?

A: You can explicitly define what the URL for each action method should be.

Q: Why does Razor Pages give the best of both worlds in routing?

A: Because Razor Pages uses conventions to generate attribute routes.

Q: What routing method does the author recommend?

A: Attribute routing is the best option and is the recommended approach.

Q: What should you still use for consistency, even with HTML-generating applications?

A: I would still stick with attribute routing in that scenario.

Q: What defines the pattern of the URL you're expecting?

A: Route templates define the pattern of the URL you’re expecting.

Q: What is a segment in a URL?

A: A segment is a small contiguous section of a URL.

Q: What can you define for each route template?

A: Specific, expected strings, variable segments, optional segments, default values, and constraints.

Q: What constraint example is mentioned for a segment of a URL?

A: Ensuring that it’s numeric.

Q: What type of route templates do Razor Pages generate by default?

A: The default convention-based attribute route templates.

Routing to Razor Pages

Keywords:

"Razor Pages uses attribute routing by creating route templates based on conventions"

"MapRazorPages in the Configure method of Startup.cs"

"the path of the Razor Page file relative to the Razor Pages root directory (Pages/)"

"creates a route template with the value 'Products/View'"

"Requests to the URL /products/view match the route template 'Products/View'"

"RoutingMiddleware selects the View\.cshtml Razor Page as the endpoint"

"Routing is not case sensitive"

"an instance of the Razor Page’s PageModel is created, and a page handler on the model is invoked"

"Razor Pages can have multiple page handlers"

"each Razor Page creates a single route template based on its file path"

"Index.cshtml pages create two route templates"

"ToDo and ToDo/Index"

"Pages/Error.cshtml, Pages/Index.cshtml, Pages/Privacy.cshtml"

"Error maps to Error.cshtml"

"Privacy maps to Privacy.cshtml"

"you want each product to have its own URL, but which map to a single Razor Page"

Questions:

Q: What kind of routing does Razor Pages use?

A: Razor Pages uses attribute routing by creating route templates based on conventions.

Q: When are route templates for Razor Pages created?

A: During app startup, when you call MapRazorPages in the Configure method of Startup.cs.

Q: How is the route template for a Razor Page file determined?

A: By the path of the Razor Page file relative to the Razor Pages root directory (Pages/), excluding the file extension.

Q: What route template is created for the Razor Page located at Pages/Products/View\.cshtml?

A: The framework creates a route template with the value "Products/View".

Q: What happens when a request is made to the URL /products/view?

A: It matches the route template "Products/View", which corresponds to the View\.cshtml Razor Page.

Q: Is routing case sensitive in ASP.NET Core?

A: Routing is not case sensitive.

Q: What does it mean when a Razor Page is executed?

A: An instance of the Razor Page's PageModel is created, and a page handler on the model is invoked.

Q: Can Razor Pages have more than one handler?

A: Razor Pages can have multiple page handlers.

Q: What is the default behavior for route templates in Razor Pages?

A: Each Razor Page creates a single route template based on its file path.

Q: What is the exception to the single route template rule in Razor Pages?

A: Index.cshtml pages create two route templates.

Q: What two route templates are created for Pages/ToDo/Index.cshtml?

A: "ToDo" and "ToDo/Index".

Q: What URLs can be used to view the Razor Page at Pages/ToDo/Index.cshtml?

A: https://example.org/ToDo or https://example.org/ToDo/Index.

Q: What Razor Pages are included in the default template created by Visual Studio or .NET CLI?

A: Pages/Error.cshtml, Pages/Index.cshtml, Pages/Privacy.cshtml.

Q: What are the route templates generated for the default Razor Pages?

A: "", "Index", "Error", "Privacy".

Q: What scenario requires more dynamic routing than the default?

A: When you want each product to have its own URL, but which map to a single Razor Page.

Customizing Razor Page route templates

Keywords:

"customize the route templates for individual pages"

"map multiple URLs to a single Razor Page"

"routing middleware parses a route template by splitting it into a number of segments"

"Each segment is either A literal value or A route parameter"

"Literal values must be matched exactly (ignoring case) by the request URL"

"each Razor Page consists of a series of literal segments"

"Literal segments in ASP.NET Core are not case-sensitive"

"route template for this page is 'About/Contact'"

"route parameters are sections of a URL that may vary but will still be a match for the template"

"parameters are required, so there must be a segment in the request URL that they correspond to"

"Some words can't be used as names for route parameters: area, action, controller, handler, and page"

"route template '{category}/{name}' could be used to match all the product-page URLs"

"value associated with the parameter is captured and stored in a dictionary of values associated with the request"

"Route values are the values extracted from a URL based on a given route template"

"Literal segments and route parameters are the two cornerstones of ASP.NET Core route templates"

Questions:

Q: What allows you to customize your application's URLs and map multiple URLs to a single Razor Page?

A: Customize the route templates for individual pages.

Q: How does the routing middleware parse a route template?

A: By splitting it into a number of segments.

Q: What are the two types of segments in a route template?

A: A literal value and a route parameter.

Q: What must literal values in a route template do to match a request URL?

A: Be matched exactly (ignoring case) by the request URL.

Q: Are literal segments in ASP.NET Core case-sensitive?

A: Literal segments in ASP.NET Core are not case-sensitive.

Q: What is the route template for the contact page located at Pages/About/Contact.cshtml?

A: About/Contact.

Q: Would the URL /about/contact/email match the route template "About/Contact"?

A: No, it would not match.

Q: How are route parameters defined in a route template?

A: By giving them a name and placing them in braces.

Q: Are route parameters optional in a route template?

A: No, the parameters are required.

Q: What are some words that can't be used as names for route parameters?

A: Area, action, controller, handler, and page.

Q: What is the route template that could match URLs like /bags/rucksack-a or /shoes/black-size9?

A: {category}/{name}.

Q: Why would the URL /socks/ not match the template "{category}/{name}"?

A: Because no name parameter is specified.

Q: Where is the value of a matched route parameter stored?

A: In a dictionary of values associated with the request.

Q: What are route values?

A: The values extracted from a URL based on a given route template.

Q: What are the two cornerstones of ASP.NET Core route templates?

A: Literal segments and route parameters.

Adding a segment to a Razor Page route template

Keywords:

"customize the Razor Page route template"

"update the @page directive at the top of the Razor Page’s .cshtml file"

"the first thing in the Razor Page file for the page to be registered correctly"

"you must include the @page directive at the top of a Razor Page’s .cshtml file"

"add an additional segment to a Razor Page’s route template"

"appends the provided route template to the default template generated for the Razor Page"

"default route template for the Razor Page at Pages/Privacy.html is 'Privacy'"

"new route template for the page would be 'Privacy/Extra'"

"add a route parameter"

"final route template of Products/{category}/{name}"

"match all of the following URLs"

"add route segments to the Razor Page template"

"use a completely custom URL for a page"

Questions:

Q: How do you customize the Razor Page route template?

A: Update the @page directive at the top of the Razor Page’s .cshtml file.

Q: What must be the first thing in a Razor Page file for the page to register correctly?

A: The @page directive.

Q: What happens if you do not include the @page directive in a Razor Page's .cshtml file?

A: ASP.NET Core will not treat the file as a Razor Page.

Q: How do you add an additional segment to a Razor Page’s route template?

A: Add a space followed by the desired route template after the @page statement.

Q: What does the @page "Extra" directive do to the default route template for Pages/Privacy.html?

A: It changes the route template to "Privacy/Extra".

Q: Where is the route template provided in the @page directive added?

A: Appended to the end of the default template for the Razor Page.

Q: What is the most common reason for customizing a Razor Page’s route template?

A: To add a route parameter.

Q: What is the final route template if the directive adds {category}/{name} to the Products page?

A: Products/{category}/{name}.

Q: What are some URLs that would match the route template Products/{category}/{name}?

A: /products/bags/white-rucksack, /products/shoes/black-size9, /Products/phones/iPhoneX.

Q: What if you don't want the /products segment in the customized URLs?

A: You can use a completely custom URL for a page.

Replacing a Razor Page route template completely

Keywords:

"stick to the default routing conventions"

"adding additional segments for route parameters"

"specify a custom route for a Razor Page"

"prefix the route with / in the @page directive"

"this is a custom route template"

"create a static custom template for a page"

"using this directive ensures it always has the route template 'checkout'"

"custom route templates that start with '/' as absolute route templates"

"relative to their location in the file hierarchy"

"the default template is no longer valid"

"the URL /Payment is no longer valid and will not execute the Razor Page"

"replaces the default route template for the page"

Questions:

Q: What should you stick to for maximum productivity when working with Razor Pages?

A: The default routing conventions.

Q: How do you specify a custom route for a Razor Page?

A: Prefix the route with / in the @page directive.

Q: What does the "/" at the start of a route in the @page directive indicate?

A: That this is a custom route template.

Q: What is the route template when using @page "/{category}/{name}"?

A: "{category}/{name}".

Q: How do you ensure a Razor Page always has the route template "checkout"?

A: Use @page "/checkout".

Q: What kind of route templates are those that start with "/"?

A: Absolute route templates.

Q: What are other route templates considered if they do not start with "/"?

A: Relative to their location in the file hierarchy.

Q: What happens to the default route template when you use a custom route?

A: The default template is no longer valid.

Q: If you use the "checkout" route template on a Razor Page at Pages/Payment.cshtml, what URL can access it?

A: /checkout.

Q: What URL will no longer work if a custom route is applied to Pages/Payment.cshtml?

A: /Payment.

Exploring the route template syntax

(Using optional and default values)

Keywords:

basic elements of literals and route parameter segments

optional URL segments

default values when a segment isn’t specified

place additional constraints on the values

optional and default values

literal segment and two required routing parameters

default value specified for it using =all

optional route parameter called id

model binding

optional parameter (that doesn’t have a default) at the end of a route template

using default values allows you to have multiple ways to call the same URL

shorter and more memorable URLs

Questions:

Q: What extra features do route templates have beyond basic literals and route parameter segments?

A: They let you have optional URL segments, provide default values when a segment isn’t specified, or place additional constraints on the values valid for a given route parameter.

Q: What happens if the URL doesn’t contain a segment corresponding to the {name} parameter?

A: The router will use the default value all instead.

Q: What does the {id?} segment in the route template represent?

A: It defines an optional route parameter called id.

Q: Can you specify a value for an optional {id} parameter without specifying {category} and {name}?

A: No, you cannot specify a value for the optional {id} parameter without also specifying the {category} and {name} parameters.

Q: Where can an optional parameter without a default value be placed in a route template?

A: It can only be placed at the end of a route template.

Q: Why would using default values in route templates be desirable?

A: Using default values allows you to have multiple ways to call the same URL and create shorter, more memorable URLs while matching a variety of routes in a single template.

Q: What are the route values for the URLs /product/shoes and /product/shoes/all?

A: Both URLs have the route values of category=shoes and name=all.

Adding additional constraints to route parameters

Keywords:

route parameter is required or optional

default value

Routing only matches URL segments to route parameters

template similar to that in figure 5.7, "{category}/{name=all}/{id?}"

router happily assigns route values and matches the template

route values assigned

model binding

OnGet(int id)

add additional constraints to a route template

{id\:int}

IntRouteConstraint

route values are convertible to appropriate types

combine multiple constraints by separating the constraints with colons

Using constraints allows you to narrow down the URLs

route template isn't considered a match

Don't use route constraints to validate general input

Constraints are best used sparingly

catch-all parameter

Questions:

Q: What does routing match when assigning route parameters?

A: Routing only matches URL segments to route parameters.

Q: What problem can occur if the id route parameter is assigned a non-integer value?

A: You'll get an exception when it's bound to the integer id parameter.

Q: How can you ensure a route parameter only matches certain types of values?

A: By adding additional constraints to a route template using a colon, for example, {id\:int}.

Q: What happens if a route template's constraints are not valid when matching a URL?

A: The route template isn't considered a match, and the Razor Page won't be executed.

Q: Why should route constraints not be used to validate general input like email addresses?

A: Doing so will result in 404 "Page not found" errors, which will be confusing for the user.

Q: How can multiple constraints be applied to a single route parameter?

A: By separating the constraints with colons.

Q: When are constraints particularly useful in routing?

A: They are useful when you have strict requirements on the URLs used by the application.

Matching arbitrary URLs with the catch-all parameter

Keywords:

route templates take URL segments and attempt to match them to parameters or literal strings

route parameters themselves won't contain a slash

URLs for this page should contain all the currencies as separate segments

catch-all parameter in the @page directive

Catch-all parameters can be declared using either one or two asterisks inside the parameter definition

match the remaining unmatched portion of a URL, including any slashes

value of the route value others would be the single string "GBP/EUR"

Catch-all parameters are greedy and will capture the whole unmatched portion of a URL

one- and two-asterisk versions of the catch-all parameter behave identically when routing

the one-asterisk version URL encoded forward slashes

the two-asterisk version doesn't encode forward slashes

mapping URLs to Razor Pages is only half of the responsibilities of the routing system

routing system in ASP.NET Core is also used to generate URLs

Questions:

Q: What character normally splits URL segments in route templates?

A: The slash character, /.

Q: How can you capture any number of URL segments after a certain point in a route template?

A: By using a catch-all parameter in the @page directive.

Q: How are catch-all parameters declared in route templates?

A: Using either one or two asterisks inside the parameter definition, like {*others} or {**others}.

Q: What value would the route value others have for the URL /USD/convert/GBP/EUR?

A: The single string "GBP/EUR".

Q: How do the one- and two-asterisk versions of catch-all parameters differ when generating URLs?

A: The one-asterisk version URL encoded forward slashes, and the two-asterisk version doesn't.

Q: Besides mapping URLs to Razor Pages, what is another responsibility of the routing system in ASP.NET Core?

A: It is also used to generate URLs so that you can easily reference your Razor Pages from other parts of your application.

Generating URLs from route parameters

Keywords:
generating URLs

redirect URLs as a response from your Razor Pages

URLs can be somewhat fluid

renaming the Pages/Cart.cshtml page to Pages/Basket/View\.cshtml

URL you use to access the page to change from /Cart to /Basket/View

manually manage these links within your app would be a recipe for heartache

routing infrastructure to generate appropriate URLs dynamically at runtime

mapping a URL to a Razor Page

routing middleware takes a URL, matches it to a route template, and splits it into route values

generator takes in the route values and combines them with a route template to build a URL

Questions:

Q: What is one way to automatically send redirect URLs from Razor Pages?

A: By generating URLs as a string you can use in your code.

Q: What happens to the URL if you rename a Razor Page like Pages/Cart.cshtml to Pages/Basket/View\.cshtml?

A: The URL you use to access the page will change from /Cart to /Basket/View.

Q: Why is manually managing hardcoded URLs within an app problematic?

A: Because you'd have to remember to do a find-and-replace with every rename, causing broken links and 404s.

Q: How does the routing infrastructure help with URLs in an application?

A: It generates appropriate URLs dynamically at runtime, freeing you from manually managing links.

Q: What is the conceptual difference between routing and URL generation?

A: Routing takes a URL, matches it to a route template, and splits it into route values; URL generation takes route values and combines them with a route template to build a URL.

Generating URLs for a Razor Page

Keywords:

generate a link to the Pages/Currency/View\.cshtml Razor Page

Url helper from the PageModel base class

Url property is an instance of IUrlHelper

Page method to which you pass the name of the Razor Page and any additional route data

route data is packaged as key-value pairs into a single C# anonymous object

provide the relative file path to the Razor Page

provide the absolute file path (relative to the Pages folder)

IUrlHelper has several different overloads of the Page method

passed in an anonymous object, new { code = "USD" }

route value will be used in the URL path, giving /Currency/View/GBP

route value is appended as additional data as part of the query string, for example /Currency/View?code=GBP

Generating URLs based on the page you want to execute is convenient

process is much the same as for Razor Pages in MVC controllers

Questions:

Q: How can you generate a link to the Pages/Currency/View\.cshtml Razor Page in your code?

A: By using the Url helper from the PageModel base class.

Q: What does the Url property represent?

A: It is an instance of IUrlHelper that allows you to easily generate URLs by referencing Razor Pages by their file path.

Q: How is route data passed to the Page method of IUrlHelper?

A: As key-value pairs packaged into a single C# anonymous object.

Q: How can you specify the path to the Razor Page when generating a URL?

A: You can provide either the relative file path or the absolute file path starting with a "/".

Q: What happens if the route template explicitly includes a route value like {code}?

A: The route value will be used in the URL path, for example /Currency/View/GBP.

Q: How is a route value handled if it is not explicitly included in the route template?

A: It is appended as additional data in the query string, for example /Currency/View?code=GBP.

Q: Is the process of generating URLs in MVC controllers different from Razor Pages?

A: The process is much the same, though the methods are slightly different.

Generating URLs for an MVC controller

Keywords:

Generating URLs for MVC controllers

use the Action method on the IUrlHelper

provide an MVC controller name and action name instead of a page path

call the Action and Page methods on IUrlHelper from both Razor Pages and MVC controllers

If the URL you need refers to a Razor Page, use the Page method

If the destination is an MVC action, use the Action method

use the C# 6 name of operator to make the value refactor-safe

use a different overload of Action that omits the controller name

Ambient values are the route values for the current request

Generating URLs using the Url property doesn’t tend to be very common in practice

more common to generate URLs implicitly with an ActionResult

Questions:

Q: What method do you use on IUrlHelper to generate URLs for MVC controllers?

A: You use the Action method on the IUrlHelper.

Q: When should you use the Page method versus the Action method on IUrlHelper?

A: Use the Page method if the URL refers to a Razor Page; use the Action method if the destination is an MVC action.

Q: What is the benefit of using the C# 6 name of operator when specifying action method names?

A: It makes the value refactor-safe.

Q: What are ambient values in routing?

A: Ambient values are the route values for the current request, including controller and action.

Q: Is generating URLs using the Url property common in practice?

A: No, it is more common to generate URLs implicitly with an ActionResult.

Generating URLs with ActionResults

(Generating a redirect URL from an ActionResult)

Keywords:

generate a string containing a URL for both Razor Pages and MVC actions

automatically redirect a user to a URL

RedirectToPage method takes the path to a Razor Page and any required route parameters

RedirectToAction to automatically redirect to an MVC action

RedirectToAction is only necessary if you're using MVC controllers to generate HTML

use Razor Pages instead of MVC controllers for HTML generation

generate URLs when building HTML in your views

generate URLs from parts of your application outside of the Razor Page or MVC infrastructure

use the LinkGenerator class

Questions:

Q: When would you use an ActionResult for URL generation instead of displaying the URL as a string?

A: You use an ActionResult to automatically redirect a user to a URL.

Q: What does the RedirectToPage method do?

A: It takes the path to a Razor Page and any required route parameters and generates a URL that automatically redirects the user.

Q: When should you use RedirectToAction instead of RedirectToPage?

A: RedirectToAction is necessary if you're using MVC controllers to generate HTML instead of Razor Pages.

Q: What is recommended for HTML generation, Razor Pages or MVC controllers?

A: It is recommended to use Razor Pages instead of MVC controllers for HTML generation.

Q: What class should you use to generate URLs outside of the Razor Page or MVC infrastructure?

A: You should use the LinkGenerator class.

Generating URLs from other parts of your application

Keywords:

keep your Razor Pages relatively simple

execute your application's business and domain logic in separate classes and services

URLs your application uses shouldn't be part of your domain logic

try to keep knowledge of the frontend application design out of your business logic

pattern is known generally as the Dependency Inversion principle

creating emails in a background service

LinkGenerator class lets you generate that URL

LinkGenerator class is available in any part of your application

IUrlHelper is typically easier and hides some details of using the LinkGenerator

LinkGenerator has various methods for generating URLs, such as GetPathByPage, GetPathByAction, and GetUriByPage

make sure you provide the correct Razor Page path and any necessary route parameters

the URL generated will be null

checking the generated URL for null explicitly

how incoming requests are routed to Razor Pages

where page handlers come into it

Questions:

Q: What should you try to keep your Razor Pages?

A: You should try to keep your Razor Pages relatively simple.

Q: Where should you execute your application's business and domain logic?

A: In separate classes and services.

Q: Why shouldn't URLs be part of your domain logic?

A: Because it makes it easier for your application to evolve or change completely.

Q: What principle advises keeping knowledge of the frontend application design out of your business logic?

A: The Dependency Inversion principle.

Q: When might you need to include a link to your application in a background service?

A: When you're creating emails in a background service.

Q: What class lets you generate URLs that update automatically if the routes in your application change?

A: The LinkGenerator class.

Q: Where is the LinkGenerator class available for use?

A: In any part of your application, including middleware and other services.

Q: Which helper is typically easier and hides some details of using the LinkGenerator?

A: The IUrlHelper.

Q: Name some methods the LinkGenerator has for generating URLs.

A: GetPathByPage, GetPathByAction, and GetUriByPage.

Q: What happens if you provide an incorrect Razor Page path or forget a required route parameter?

A: The URL generated will be null.

Q: What is recommended to ensure there are no problems with URL generation?

A: Checking the generated URL for null explicitly.

Q: What topic will the next section cover after routing incoming requests to Razor Pages?

A: Page handlers and how you can have multiple handlers on a Razor Page.

Selecting a page handler to invoke

Keywords:

routing was about mapping URLs to a handler

page handler

EndpointMiddleware selects which page handler to invoke

Razor Pages can have multiple handlers

RoutingMiddleware selects a Razor Page

EndpointMiddleware needs to know how to choose which handler to execute

The HTTP verb used in the request

The value of the handler route value

handler route value typically comes from a query string value in the request URL

include the {handler} route parameter in your Razor Page's route template

EndpointMiddleware uses the handler route value and the HTTP verb together with a standard naming convention

async suffix is also optional

OnGet—Invoked for GET requests that don't specify a handler value

OnPostAsync—Invoked for POST requests that don't specify a handler value

OnPostCustomSearch—Invoked for POST requests that specify a handler value of "CustomSearch"

EndpointMiddleware executes an implicit page handler instead

implicit page handlers take part in model binding and use page filters but execute no logic

if a request uses the HEAD verb, and there is no corresponding OnHead handler, Razor Pages will execute the OnGet handler instead

Questions:

Q: What does routing mean for Razor Pages?

A: Routing means mapping URLs to a handler, which is a page handler for Razor Pages.

Q: What does the EndpointMiddleware do when it executes a Razor Page?

A: It selects which page handler to invoke.

Q: How many page handlers can a Razor Page have and how many run per request?

A: A Razor Page can have multiple handlers, but only one runs in response to a given request.

Q: What two variables does the EndpointMiddleware use to select a page handler?

A: The HTTP verb used in the request and the value of the handler route value.

Q: Where does the handler route value typically come from in the request URL?

A: It typically comes from a query string value.

Q: How can you avoid using query strings for the handler route value?

A: By including the {handler} route parameter in your Razor Page's route template.

Q: What naming convention does the EndpointMiddleware use to identify which page handler to execute?

A: It uses the handler route value and the HTTP verb together with a standard naming convention.

Q: Which page handler is invoked for GET requests that don't specify a handler value?

A: OnGet.

Q: Which page handler is invoked for POST requests that don't specify a handler value?

A: OnPost Async.

Q: Which page handler is invoked for POST requests that specify a handler value of "CustomSearch"?

A: OnPostCustomSearch.

Q: What happens if a request does not match any defined verb-handler pairs?

A: The EndpointMiddleware executes an implicit page handler instead.

Q: What do implicit page handlers do?

A: They take part in model binding and use page filters but execute no logic.

Q: What happens if a request uses the HEAD verb and there is no OnHead handler?

A: Razor Pages will execute the OnGet handler instead.

Customizing conventions with Razor Pages

(Configuring routing conventions using RouteOptions in Startup.cs)

Keywords:

Razor Pages is built on a series of conventions

customize those conventions

full control over your application’s URLs

match the filenames of your Razor Pages very closely

route template Products/Product-Details

words in URLs are usually separated using “kebab-case” rather than “PascalCase”

URLs always end with a trailing slash

change these conventions by configuring a RouteOptions object in Startup.cs

configuration for the whole ASP.NET Core routing infrastructure

create a custom parameter transformer

regular expression to replace PascalCase values in a generated URL with kebab-case

register the parameter transformer in your application with the AddRazorPagesOptions extension method

completely customize the conventions used by Razor Pages

add an extra page route convention for a given Razor Page

Questions:

Q: What is Razor Pages built on to reduce boilerplate code?

A: Razor Pages is built on a series of conventions.

Q: How closely do default ASP.NET Core URLs match Razor Page filenames?

A: They match the filenames of your Razor Pages very closely.

Q: What URL naming style is preferred over PascalCase according to the passage?

A: Kebab-case is preferred over PascalCase.

Q: What URL formatting is commonly ensured at the end of URLs?

A: URLs always end with a trailing slash.

Q: How can you change the default routing conventions for Razor Pages and MVC?

A: By configuring a RouteOptions object in Startup.cs.

Q: What must you create to use kebab-case URLs in your application?

A: You must create a custom parameter transformer.

Q: How does the parameter transformer convert PascalCase to kebab-case?

A: It uses a regular expression to replace PascalCase values in a generated URL with kebab-case.

Q: How do you register the custom parameter transformer in your application?

A: By using the AddRazorPagesOptions extension method in Startup.cs.

Q: What else can you add using AddRazorPagesOptions besides registering the parameter transformer?

A: You can add an extra page route convention for a given Razor Page.

Registering a parameter transformer using RazorPagesOptions

Keywords:

The AddPageRoute convention adds an alternative way to execute a single Razor Page

using AddPageRoute adds an extra route template to the page instead of replacing the default

customize all the pages in your application

custom convention

Microsoft's "Razor Pages route and app conventions in ASP.NET Core" documentation

Conventions are a key feature of Razor Pages

Avoid replacing the route template with an absolute path in a page's @page directive

Avoid adding literal segments to the @page directive

Avoid adding additional route templates to a Razor Page with the AddPageRoute convention

Do add route parameters to the @page directive to make your routes dynamic

Do consider using global conventions

stick to the conventions

overlapping route templates

exception at runtime when your application receives a request that matches multiple route templates

Routing is one of those topics that people often get stuck on

We'll revisit routing again when I describe how to create Web APIs in chapter 9

In chapter 6 we'll dive into model binding

route values generated during routing are bound to your action method or page handler parameters

Questions:

Q: What does the AddPageRoute convention do for a Razor Page?

A: It adds an extra route template to the page instead of replacing the default.

Q: What should you avoid when customizing route templates using the @page directive?

A: Avoid replacing the route template with an absolute path and avoid adding literal segments.

Q: When is it advised to use global conventions?

A: When you want to change the route templates for all your Razor Pages.

Q: What is the risk of creating overlapping route templates in Razor Pages?

A: You won’t get an error at compile time, but you’ll get an exception at runtime when a request matches multiple route templates.

Q: What chapter will revisit routing with regard to creating Web APIs?

A: Chapter 9.

Q: What will chapter 6 cover related to routing?

A: How route values generated during routing are bound to your action method or page handler parameters and how to validate those values.